REMARKS

In response to the Office Action mailed on November 17, 2006, Applicant(s) respectfully request(s) reconsideration.

Claim(s) 1-33 are now pending in this Application.

Claim(s) 1, 14, 17, 18, 28 and 31-33 are independent claims and the remaining claims are dependent claims.

In this Amendment, claim(s) 1, 14, 15, 17, 18, 28, 29 and 31-33 have been amended and claim(s) 4, 16, 25 and 30 have been cancelled. Applicant(s) believe that the claim(s) as presented are in condition for allowance. A notice to this affect is respectfully requested.

Rejection under 35 U.S.C. §101:

Claims 31-33 were rejected under 35 U.S.C. §101 as being nonstatutory. Claims 1-33 have been herein amended to recite a computer readable storage medium, and further that the storage medium stores a set of processor based instructions, to further clarify the statutory nature of Applicant's claimed invention.

Rejection under 35 U.S.C. §112:

Claims 15, 17-27 and 19 were rejected under 35 U.S.C. §112 as for indefiniteness. These claims have all been amended per the above to provide the proper antecedent basis.

Rejection under 35 U.S.C. §102(b) based on Whitehead, U.S. Patent No. 6,085,030:

Claim 1-11, 13-25 and 27-33 were rejected under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 6,085,030 to Whitehead, et. al. (Whitehead '030). Applicant(s) respectfully disagree(s) with these contentions and assert that the present claimed invention is not anticipated by any disclosure in the Whitehead '030 reference. Specifically, the Office Action rejects claim 1, suggesting that Whitehead '030 teaches instantiating the native object and maintaining the link between the API object and the native object, as recited in claim 1.

The Whitehead '030 reference instantiates the entire object on the remote/target node for satisfying the request. In contrast, the claimed invention instantiates only an API object in the client object space. The Office Action cites Fig. 2 of Whitehead and col. 7:48-50 in support of this assertion. Whitehead, however, instantiates a native object from the object model for providing a native service. Whitehead does not show, teach, or disclose instantiating the claimed API object because the claimed API object maintains a link providing a dynamic reflection to the native object on the base application server (page 14, lines 26-30). The maintained link provides a dynamic reflection of the native object in the API object, rather than duplicative reinstantiation of the native object, which further differs from Whitehead because the API object provides a direct reference linkage from the API object to the native object, as discussed at page 12, lines 11-15.

Accordingly, in support of these distinctions, Claim 1 has been amended with the subject matter of claim 4, to recite that maintaining the link comprises referencing, in a realtime manner, the native object in response to operations to the instantiated API object, such that the operations produce a nonduplicative, atomic result in the native object via the instantiated API object, to further clarify and distinguish Applicant's claimed invention. Claim 4 clarifies that the API object produces a nonduplicative, atomic result in the native object via the API object, as now recited in amended Claim 1, and disclosed at page 16, line 30-page 17, line 9. Amended claim 1 now further differs because Whitehead does not show, teach or disclose, alone or in combination, that maintaining the link provides a nonduplicative, atomic result in the native object via the API object, as discussed at page 17, lines 3-9. Claim 17, rejected on similar grounds, has been similarly amended.

The Office Action further suggests that Whitehead teaches the subject matter of claim 4 at col. 8, lines 28-35. Whitehead teaches, rather, determining which object to instantiate, an operation complete when the object is instantiated. Whitehead does not disclose maintaining the dynamic reflection from the API object to the native object via the direct reference linkage, as recited in amended Claim 1. Therefore, the claimed approach instantiates a reference object (API object) on the remote deployment node,

maintaining the base object on the server, while the cited reference performs the reverse approach, instantiates the actual component on the remote deployment node, pointing back to a proxy on the server node, as discussed at col. 11, line 61- col. 12, line 3.

The claimed invention solves the problem of instantiating a plurality of full-function objects across the network, thus avoiding the problem of memory and processor resources required by multiple instantiated objects. The cited reference, rather, facilitates multiple instantiations and deployment of such full-function objects, as discussed at col. 4, lines 36-52. As shown in Fig. 6, the instantiated, deployed component 630 is a full function component instantiated from the CMS 610, not an API object having only a direct reference linkage back to the counterpart native object in the server base object space, as discussed at page 12, lines 11-18. Claim 1 is therefore respectfully submitted as allowable.

Independent Claim 18 has been rejected on similar grounds. Claim 18 has been amended with the subject matter of claim 25, to clarify the augmented exposure of the deployed API object as opposed to the corresponding native object. The Whitehead object generator (240), makes no distinction between the claimed API object and the native object (and the respective object spaces). Amended claim 18 now clarifies the limited exposure provided by the API object upon instantiation, thus limiting the runtime behavior to that intended to be exposed. Amended claim 18 is distinguishable because Whitehead '030 does not show, teach, or disclose the claimed build facility operable to: identify templates corresponding to object types, the object types corresponding to operations for providing the remote API, define metadata for each of the objects for exposure in the remote API, the metadata identifying, for each of the exposed objects, runtime behavior of the object; and build, via an API object generator in the build facility, the exposed objects for invocation by the client application, therefore limiting the exposure but providing exposed operations via the dynamic links back to the native object, as discussed above, disclosed further at page 17, lines 14-22.

Independent claim 31, likewise rejected, has been amended with the subject matter of both claims 4 and 25, as discussed above, to further distinguish Applicant's claimed invention.

Independent claim 32 has been amended similarly to claim 31, and further with the subject matter of claim 26 to clarify a configuration in a storage area network (SAN) having a toolkit of entry points corresponding to the API objects for invocation by SAN agents, a feature not shown, taught or disclosed by Whitehead '030.

Independent claim 33 has been herein amended with the subject matter of claims 4 and 25, as above, and further to recite that the API objects have a direct reference linkage to counterpart native objects in the base application object space, as discussed above and at page 11, line 25-page 12, line 18, to further clarify and distinguish applicant's claimed invention.

Independent claims 14 and 28, also rejected under Whitehead '030, have been herein amended with the subject matter of claim 16 to clarify the limited exposure provided by the metadata, as discussed above with respect to claim 18. Claims 14 and 28 are respectfully submitted as allowable because Whitehead '030 makes no showing, teaching, or disclosure of metadata operable to identify object classes for the object types in the base application object model and identify, for each of the object types, attributes operable for definition in the corresponding object in the remote access object library, as now recited in amended claims 14 and 28.

As the remaining claims depend, either directly or indirectly, from claims 1, 14, 18 or 28, it is respectfully submitted that all claims are now in condition for allowance.

Applicant(s) hereby petition(s) for any extension of time which is required to maintain the pendency of this case. If there is a fee occasioned by this response, including an extension fee, that is not covered by an enclosed check, please charge any deficiency to Deposit Account No. 50-3735.

If the enclosed papers or fees are considered incomplete, the Patent Office is respectfully requested to contact the undersigned collect at (508) 616-9660, in Westborough, Massachusetts.

Respectfully submitted,

Christopher J. Lutz, Esq.
Attorney for Applicant(s)
Registration No.: 44,883
Chapin Intellectual Property Law, LLC
Westborough Office Park
1700 West Park Drive
Westborough, Massachusetts 01581
Telephone: (508) 616-9660
Facsimile: (508) 616-9661

Attorney Docket No.: EMC03-18(03088)

Dated: February 16, 2007